

## AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

### Listing of Claims:

- 1           1.       (Currently amended): A method for executing a start transactional  
2       execution (STE) instruction to facilitate transactional execution on a processor,  
3       comprising:  
4           encountering the STE instruction during execution of a program, wherein  
5       the STE instruction marks the beginning of a block of instructions to be executed  
6       transactionally; ~~and~~  
7           executing the STE instruction prior to executing the block of instructions;  
8       and  
9           ~~upon encountering wherein executing the STE instruction;~~ involves  
10      commencing transactional execution of the block of instructions following the  
11      STE instruction;  
12           wherein changes made during the transactional execution are not  
13      committed to the architectural state of the processor until the transactional  
14      execution successfully completes.
- 1           2.       (Original): The method of claim 1, wherein the STE instruction  
2       specifies an action to take if transactional execution of the block of instructions  
3       fails.
- 1           3.       (Original): The method of claim 2, wherein the action to take can  
2       include branching to a location specified by the STE instruction.

1           4.       (Original): The method of claim 2, wherein the action to take can  
2 include acquiring a lock on the block of instructions.

1           5.       (Original): The method of claim 2, wherein the action to take can  
2 include setting state information within the processor to indicate a failure during  
3 transactional execution of the block of instructions, thereby enabling other  
4 software executed by the processor to manage the failure.

1           6.       (Original): The method of claim 1, wherein if the transactional  
2 execution completes without encountering an interfering data access from another  
3 process or other type of failure, the method further comprises:  
4           atomically committing changes made during the transactional execution,  
5 and  
6           resuming normal non-transactional execution.

1           7.       (Original): The method of claim 1, wherein if an interfering data  
2 access from another process is encountered during the transactional execution, the  
3 method further comprises:  
4           discarding changes made during the transactional execution; and  
5           attempting to re-execute the block of instructions.

1           8.       (Original): The method of claim 1, wherein potentially interfering  
2 data accesses from other processes are allowed to proceed during the transactional  
3 execution of the block of instructions.

1           9.       The method of claim 1, wherein the block of instructions to be  
2 executed transactionally comprises a critical section.

1           10.     (Original): The method of claim 1, wherein commencing  
2     transactional execution of the block of instructions involves:  
3           saving the state of processor registers;  
4           configuring the processor to mark cache lines during loads that take place  
5     during transactional execution;  
6           configuring the processor to mark cache lines during stores that take place  
7     during transactional execution; and  
8           configuring the processor to continually monitor data references from  
9     other threads to detect interfering data references.

1           11.     (Original): The method of claim 1, wherein the STE instruction is  
2     a native machine code instruction of the processor.

1           12.     (Original): The method of claim 1, wherein the STE instruction is  
2     defined in a platform-independent programming language.

1           13.     (Currently amended): A computer system that supports a start  
2     transactional execution (STE) instruction to facilitate transactional execution,  
3     wherein the STE instruction marks the beginning of a block of instructions to be  
4     executed transactionally, the computer system comprising:  
5           a processor; and  
6           an execution mechanism within the processor;  
7           wherein upon encountering the STE instruction, the execution mechanism  
8     is configured to:  
9                     execute the STE instruction prior to executing the block of  
10                    instructions; and to  
11                    commence transactional execution of the block of  
12                    instructions following the STE instruction;

13            wherein changes made during the transactional execution are not  
14 committed to the architectural state of the processor until the transactional  
15 execution successfully completes.

1            14.    (Original): The computer system of claim 13, wherein the STE  
2 instruction specifies an action to take if transactional execution of the block of  
3 instructions fails.

1            15.    (Original): The computer system of claim 14, wherein the action  
2 to take can include branching to a location specified by the STE instruction.

1            16.    (Original): The computer system of claim 14, wherein the action  
2 to take can include acquiring a lock on the block of instructions.

1            17.    (Original): The computer system of claim 14, wherein the action  
2 to take can include setting state information within the processor to indicate a  
3 failure during transactional execution of the block of instructions, thereby  
4 enabling other software executed by the processor to manage the failure.

1            18.    (Original): The computer system of claim 13, wherein if the  
2 transactional execution completes without encountering an interfering data access  
3 from another process or other type of failure, the execution mechanism is  
4 configured to:  
5            atomically commit changes made during the transactional execution, and  
6 to  
7            resume normal non-transactional execution.

1           19.     (Original): The computer system of claim 13, wherein if an  
2     interfering data access from another process is encountered during the  
3     transactional execution, the execution mechanism is configured to:  
4           discard changes made during the transactional execution; and to  
5           attempt to re-execute the block of instructions.

1           20.     (Original): The computer system of claim 13, wherein the  
2     computer system is configured to allow potentially interfering data accesses from  
3     other processes to proceed during the transactional execution of the block of  
4     instructions.

1           21.     (Original): The computer system of claim 13, wherein the block of  
2     instructions to be executed transactionally comprises a critical section.

1           22.     (Original): The computer system of claim 13, wherein while  
2     commencing transactional execution of the block of instructions, the execution  
3     mechanism is configured to:  
4           save the state of processor registers;  
5           configure the processor to mark cache lines during loads that take place  
6     during transactional execution;  
7           configure the processor to mark cache lines during stores that take place  
8     during transactional execution; and to  
9           configure the processor to continually monitor data references from other  
10    threads to detect interfering data references.

1           23.     (Original): The computer system of claim 13, wherein the STE  
2     instruction is a native machine code instruction of the processor.

1           24.     (Original): The computer system of claim 13, wherein the STE  
2     instruction is defined in a platform-independent programming language.

1           25.     (Currently amended): A computing means that supports a start  
2     transactional execution (STE) instruction to facilitate transactional execution,  
3     wherein the STE instruction marks the beginning of a block of instructions to be  
4     executed transactionally, comprising:  
5             a processing means; and  
6             an execution means within the processing means;  
7                     wherein upon encountering the STE instruction, the  
8             execution means is configured to:  
9                     execute the STE instruction prior to executing the block of  
10            instructions; and to  
11                     commence transactional execution of the block of  
12            instructions following the STE instruction;  
13            wherein changes made during the transactional execution are not  
14     committed to the architectural state of the processor until the transactional  
15     execution successfully completes.